A Software Architecture for Automatic Deployment of CORBA Components Using Grid Technologies

Sébastien Lacour, Christian Pérez, Thierry Priol

A Software Architecture for Automatic Deployment of CORBA Components Using Grid Technologies

Sébastien Lacour — Christian Pérez — Thierry Priol

IRISA / INRIA
Campus de Beaulieu
35042 Rennes, France
{Sebastien.Lacour,Christian.Perez,Thierry.Priol}@irisa.fr

ABSTRACT. *Software components turn out to be a convenient model to build complex applications for scientific computing and to run them on a computational grid. However, deploying complex, component-based applications in a grid environment is particularly arduous. To prevent the user from directly dealing with a large number of execution hosts and their heterogeneity within a grid, the application deployment phase must be as automatic as possible. This paper describes an architecture for automatic deployment of component-based applications on computational grids. In the context of the* CORBA *Component Model (*CCM*), this paper details all the steps to achieve an automatic deployment of components as well as the entities involved: a grid access middleware and its grid information service (like OGSI), a component deployment model, as specified by* CCM*, an enriched application description and a deployment planner in order to select resources and map components onto computers.*

RÉSUMÉ. *Les composants logiciels sont une solution bien adaptée pour construire des applications complexes de calcul scientifique destinées à être exécutées sur une grille de calcul. Cependant, le déploiement d'applications complexes à base de composants sur une grille est une tâche particulièrement ardue. Pour éviter d'avoir à faire face directement au grand nombre d'ordinateurs de la grille et à leur hétérogénéité, la phase de déploiement d'application doit être automatisée. Cet article décrit une architecture de déploiement automatique d'applications à base de composants sur grille de calcul. En partant du modèle de composants* CORBA *(*CCM*), ce papier détaille les étapes du déploiement de composants et les acteurs en présence : un intergiciel d'accès aux ressources de la grille (à l'instar de OGSI), un modèle de déploiement de composants, une description étendue de l'application et un planificateur de déploiement.*

KEYWORDS: CORBA *Components, Automatic Deployment, Computational Grids.*

MOTS-CLÉS : *Composants* CORBA*, déploiement automatique, grilles de calcul.*

1. Introduction

Modern software development approaches are often suspected of not providing the level of performance which high-end parallel computers would offer. However, new scientific applications become more and more complex, involving several simulation codes coupled together to obtain more accurate simulations. For example, multi-physics simulations aim to simulate various physics, each of them implemented by a dedicated code, to increase the accuracy of simulation. It is becoming clear that a radical shift in software development should occur to handle the increasing complexity of such applications. Moreover, the computing infrastructure should provide the level of performance to running such applications within a reasonable time frame. A computational grid is by no doubt a computing infrastructure that could deliver this level of performance. It is a set of high-performance computing resources connected to the Internet and managed by a middleware that gives transparent access to resources wherever should they be located in the network.

Software components turn out to be a convenient model to build multi-physics applications for scientific computing and to run them on a computational grid [PéR 03, ARM 99]. Each simulation code can be encapsulated into a component. Such an approach raises several difficult problems such as encapsulation of parallel simulation codes into software components and efficient communication between components in the presence of various high-performance networking technologies. We already proposed solutions [PéR 03, DEN 03] to those problems in the context of the CORBA component model (CCM) [Obj 02]. For better acceptance of component-based applications running on a grid, the deployment phase should be as automatic as possible while taking into account application constraints (memory, *etc.*) and/or user constraints. While environments like ProActive [BAU 02] are able to deal with Grid middleware, they do not support application and/or user constraints: the mapping of virtual nodes to physical nodes has to be provided manually and network constraints seem difficult to handle.

This paper presents an architecture for automatic deployment of component-based applications on computational grids. Section 2 details all the necessary entities and their relationships to achieve an automatic deployment of components. Examples of these entities are presented with respect to the prototype we are currently developing. Before the conclusion, Section 3 gives an overview of the upcoming challenges.

2. Architecture for Automatic Deployment of Components

The CORBA component model contains a deployment model that specifies how a particular component can be installed, configured and launched on a machine. The specifications do not deal with the problem of selecting machines and rely on a Server-Activator daemon to actually launch component servers.

The proposed architecture aims to describe the entities needed for an automatic deployment as well as their relationships. These entities can be grouped into three

parts, each of them actually corresponding to a phase of the deployment process: the inputs (the component assembly and a grid resource description), the planner, which selects the resources and maps each component on a computer, and the actual deployment of the components on the selected resources. This section reviews these entities and mentions a few issues which have already been tackled within our prototype.

2.1. *Information Description*

Two pieces of information are required for automatic deployment: a description of the component-based application to deploy and a description of the grid resources on which the application may be deployed.

2.1.1. *Component-Based Application Description*

Within the context of the CORBA Component Model (CCM, [Obj 02]), an application is made of a set of components, called a component assembly package. It is an archive provided by the user to the deployment tool. It includes, among other files, the assembly description which describes all the components of the assembly and their interconnections.

The assembly and component descriptors can express various requirements such as the processor architecture and the operating system required by a component implementation. A component may have environmental or other dependencies, like libraries, executables, JAVA classes, *etc*. Another possible requirement is component collocation: components may be free or partitioned to a single process or a single host, meaning that a group of component instances will have to be deployed in the same process or on the same compute node.

2.1.2. *Grid Resources*

Information about grid resources includes not only compute and storage resource information, but also network description. While compute and storage resource description is rather well mastered (computer architecture, number and speed of CPUs, operating system, memory size, storage capacity, *etc*.), network description received less attention. We have proposed [LAC 04b] a scalable model for grid network topology description and have implemented it on top of MDS2 [CZA 01], the information service of the Globus Toolkit [Glo] version 2.

The deployment tool requires a pointer to a resource information service to be able to automatically find adequate resources. Depending on the type of resource information service, the pointer can be a path to a local file, a URL, or a distinguished name (DN), host and port to access an LDAP tree. Our prototype [LAC 04b] currently supports local file access, HTTP(S) and (GSI)FTP protocols as well as LDAP / MDS2 query.

2.2. *Deployment Planning*

The deployment planner is responsible for 1) selecting the computers which will run the components and the component servers, 2) selecting the network links (or network technology) to interconnect the components, and 3) mapping the component servers onto the selected computers. The input of the deployment planning algorithm is made of the application description and the resource description, as explained in Subsection 2.1.

The output of the deployment planner is a deployment plan that describes the mapping of the components onto component servers and the mapping of these component servers onto the computers of the grid. The deployment plan should also specify 1) in what order processes must be launched by the deployment tool, 2) how data must flow from the output of certain processes to the input of other processes, 3) what network connections must be established between every pair of processes. For instance, items 1) and 2) are necessary for CORBA applications, where a Naming Service needs to be launched, and its reference needs to be passed to the other processes.

Our prototype is currently based on a simple round-robin deployment planning algorithm. It is just a proof of concept.

2.3. *Actually Launching Components on a Grid*

Once the deployment plan has been obtained from the previous step, the component-based application is launched and configured according to the CORBA component model. The technical point is that the selected machines are assumed not to contain any component activator or component server. That is why a job submission method is needed to launch this very first process. This step is fully compatible with the CCM deployment model [LAC 04a] but needs more work to comply with the MDA deployment specification [MDA]. The deployment tool manages two sorts of handles: CORBA references and handles returned by the grid access middleware. Both are useful to control application processes, like cancel, suspend, or restart their execution.

To face the diversity of grids, the deployment tool should support various grid access middleware such as the Globus Toolkit [CZA 98], OSGA, Condor [FRE 01], *etc.* Our prototype illustrates how CORBA components can be deployed on a computational grid using the Globus Toolkit [Glo]: more details are provided in [LAC 04a].

3. Efficient Automatic Deployment

While a few issues have already been addressed as mentioned in Section 2, there remains a number of issues to achieve an efficient automatic deployment which are mainly related to the constraints attached to an application. The central issue is to understand what a user expects from an automatic deployment tool and what is possible, like prediction of the behavior of a component [FUR 02] for example.

3.1. *Enriching the Application Description*

The constraints attached to a component assembly package mainly focus on enabling the execution of the component. New kinds of constraints could be useful, like communication requirements (latency, bandwidth, *etc.*) or a description of the behavior of the application with respect to specific resources [FUR 02]. CCM specifications allow new constraints to be added to the component assembly package as in [WAN 03] for example. A major issue is to standardize useful constraints.

3.2. *Taking User-Level Constraints into Account*

The deployment planning algorithm (see Subsection 2.2) requires a *goal* [KIC 04] to produce a deployment plan. For example, do we want to minimize the execution time or do we want the application to run at a particular site, close to a visualization node? Those constraints are not specific to the application itself, they are user-level constraints. They belong neither to the application description nor to the grid resource description. To take them into account, a third kind of information needs to be defined.

3.3. *Deployment Planning Algorithm*

The deployment planning algorithm of our prototype is too simple to satisfy the constraints mentioned above.. More sophisticated algorithms like Sekitei [KIC 04] exist, but the question is to determine if they are suitable for our purpose. To support a variety of application-level and user-level constraints, the planner needs to be very customizable. Do we need a general purpose deployment planning algorithm? Or do we need a collection of specialized algorithms? The latter solution may give better results, since we can imagine that an application may provide its own fine-tuned deployment algorithm.

4. Conclusion

On the one hand, software component technologies appear to be a convenient model to handle the complexity of multi-physics simulations. On the other hand, grids promise to offer the necessary level of performance for such applications. This paper has presented an architecture to achieve automatic deployment of component-based applications in a grid environment. The central entity is the deployment planner which has to select resources and map components on them to achieve a goal. The planner requires the description of both the component assembly and grid resources. It generates a deployment plan which controls the CCM deployment with the help of a job submission method. Remaining issues of our ongoing work include the definition of useful application-level constraints, management of user-level constraints, and integration of efficient deployment planning algorithms.

5. References

[ARM 99] ARMSTRONG R., GANNON D., GEIST A., KEAHEY K., KOHN S., MCINNES L., PARKER S., SMOLINSKI B., "Toward a Common Component Architecture for High-Performance Scientific Computing", *Proc. of the 8th Intl. Symp. on High Performance Distributed Computation*, Aug. 1999.

[BAU 02] BAUDE F., CAROMEL D., MESTRE L., HUET F., VAYSSIÈRE J., "Interactive and Descriptor-based Deployment of Object-Oriented Grid Applications", *Proc. of the 11th Intl. Symp. on High Performance Distributed Computing*, July 2002, p. 93-102.

[CZA 98] CZAJKOWSKI K., FOSTER I., KARONIS N., KESSELMAN C., MARTIN S., SMITH W., TUECKE S., "A Resource Management Architecture for Metacomputing Systems", *Proc. of the Workshop on Job Scheduling Strategies for Parallel Processing*, vol. 1459 of *LNCS*, 1998, p. 62-82.

[CZA 01] CZAJKOWSKI K., FITZGERALD S., FOSTER I., KESSELMAN C., "Grid Information Services for Distributed Resource Sharing", *Proc. of the 10th IEEE Intl. Symp. on High-Performance Distributed Computing*, San Francisco, CA, Aug. 2001, p. 181-194.

[DEN 03] DENIS A., PÉREZ C., PRIOL T., "PadicoTM: An Open Integration Framework for Communication Middleware and Runtimes", *FGCS*, vol. 19, 2003, p. 575-585.

[FRE 01] FREY J., TANNENBAUM T., LIVNY M., FOSTER I., TUECKE S., "Condor-G: A Computation Management Agent for Multi-Institutional Grids", *Proc. of the 10th Intl. Symp. on High Performance Distributed Computing (HPDC)*, Aug. 2001, p. 55-63.

[FUR 02] FURMENTO N., MAYER A., MCGOUGH S., NEWHOUSE S., FIELD T., DARLINGTON J., "ICENI: Optimisation of Component Applications within a Grid Environment", *Journal of Parallel Computing*, vol. 28, num. 12, 2002, p. 1753-1772.

[Glo] The Globus Alliance: `http://www.globus.org/`.

[KIC 04] KICHKAYLO T., KARAMCHETI V., "Optimal Resource-Aware Deployment Planning for Component-based Distributed Applications", *Proc. of the 13th International Symp. on High Performance Distributed Computing (HPDC)*, Honolulu, HI, June 2004.

[LAC 04a] LACOUR S., PÉREZ C., PRIOL T., "Deploying CORBA Components on a Computational Grid: General Principles and Early Experiments Using the Globus Toolkit", *Proc. of the 2nd Intl. Conf. on Component Deployment*, vol. 3083 of *LNCS*, 2004, p. 35-49.

[LAC 04b] LACOUR S., PÉREZ C., PRIOL T., "A Network Topology Description Model for Grid Application Deployment", Research Report num. RR-5221, June 2004, INRIA, IRISA, Rennes, France, available at `http://www.inria.fr/rrrt/rr-5221.html`.

[MDA] Deployment and Configuration of Component-based Distributed Applications Specification: `http://www.omg.org/cgi-bin/doc?ptc/2003-07-02`.

[Obj 02] OBJECT MANAGEMENT GROUP (OMG), "CORBA Components, Version 3", Document num. formal/02-06-65, June 2002.

[PéR 03] PÉREZ C., PRIOL T., RIBES A., "A Parallel CORBA Component Model for Numerical Code Coupling", *The International Journal of High Performance Computing Applications (IJHPCA)*, vol. 17, num. 4, 2003, p. 417-429, SAGE Publications.

[WAN 03] WANG N., RODRIGUES C., GILL C., "A QoS-aware CORBA Component Model for Distributed, Real-time, and Embedded System Development", *OMG Workshop On Embedded & Real-Time Distributed Object Systems*, Washington D.C., Jul. 2003.